D3D System Requirements
Installation and Configuration Guide

D3D System Requirements

1.1 Hardware requirements

1.1.1 Platform

A PC/Linux system with a **500MHz** (or faster) Pentium (or equivalent) is recommended. Since D3D makes no explicit use of multiple processors, they are optional. However, the operating system *may* take advantage of multiple processors, and this *may*, in turn, improve performance.

1.1.2 Memory

256MB is the recommended minimum, although D3D has worked on systems with only 128MB. The maximum amount of memory that D3D will attempt to use can be limited (see section 2.5.1).

1.1.3 Graphics adapter

At this time, D3D supports software rendering only. As a result, rendering speed is directly related to CPU speed. Accelerated, or hardware, rendering may require special device drivers, X servers, or even modified OS kernels. This makes accelerated rendering very difficult for FSL to evaluate, implement, document, and support. However, with the advent of newer X servers, support for accelerated rendering may be possible in the future.

1.1.4 Disk space

D3D application: 20MB (directory **d3d_demo**)

AWIPS localization data: 20MB (directory **awips**)

The awips directory on the D3D CD contains a subset of the AWIPS D2D localization data should be sufficient for systems that do not have an existing AWIPS D2D localization data directory. On systems with access to real-time D2D data, it may be preferable to use the localization data already in place on those systems.

Documentation: 60MB (directory **docs**)

The documentation does not need to installed for D3D to run. However, if it is not installed, the on-line user's guide will not be available through D3D's **Help** option. This documentation is also available at **http://d3d.fsl.noaa.gov**.

Case data: 560MB (directory **case)**

Case data is provided on the D3D CD solely for demonstration and evaluation purposes. It may be used from the CD, or installed on the user's system. On systems with access to real-time D2D data, it may be preferable to use that data.

1.2 Software requirements

1.2.1 Operating system

Red Hat Linux **6.2** (or later).

1.2.2 Tcl/Tk

Tool Command Language/Tool Kit version **8.0.5** (or later). This is the shareware software that runs **wish** and the D3D GUI.

1.2.3 AWIPS D2D data and localization files

AWIPS D2D is **not** required to run D3D. However, D3D has been linked with D2D data access functions. These require that data must be formatted and organized the same as for D2D and that D2D localization data be available.

D3D Installation and Configuration

2.1 Installation

1) Determine the directory where D3D is to be installed and copy the CD directory **d3d_demo** (and all of its contents) to that location. **d3d_demo** is the D3D main directory and is referred to as $D3D_DIR.

2) Locate the AWIPS D2D localization data directory, which is typically **/awips/fxa**. If no localization data directory exists, copy the CD directory **awips** to a suitable location where **../awips/fxa** will be referred to as $FXA_HOME.

3) Locate the AWIPS D2D data directory, which is typically **/data/fxa** and is referred to as $FXA_DATA. If a D2D data directory isn't available, the case data provided on the D3D CD may be used instead. This case data is located in the **case** directory and may be accessed from the CD or copied to disk and used from there. When case data is being used, the location of the **case** directory is referred to as $FXA_CASE.

2.2 Directory structure and organization

$D3D_DIR	D3D main directory
$D3D_DIR/app	Scripts, source code, binary files, and configuration files for D3D. Referred to as $D3D_HOME.
$D3D_DIR/localization	D3D localization data files, the most noteworthy of which is d3denv because it sets the environmental variables that D3D uses.
$D3D_DIR/user_files	By default, bundle and procedure (.sav) files are written to this directory. As distributed, this directory contains a few samples of bundles.
$D3D_DIR/logs	If this directory exists and is writable, log files are automatically written here. Log files record D3D activity and can be used to track usage of the application. *NOTE*: $D3D_DIR/logs is **not** created during installation. It must be created by the installer if logging is desired.

2.3 Environment variables

Environment variables used by D3D are defined in $D3D_DIR/localization/**d3denv**. The values shown below are examples for an FSL localization. These values may need to be changed to reflect the AWIPS D2D localization and D3D directory structure being used on your system.

2.3.1 AWIPS variables (when using real-time data)

```
FXA_HOME=/awips/fxa
FXA_DATA=/data/fxa
FXA_CASE=
FXA_LOCAL_SITE=FSL
FXA_INGEST_SITE=FSL
FXA_NATL_CONFIG_DATA=${FXA_HOME}/data/localization
FXA_LOCALIZATION_ROOT=${FXA_HOME}/data/localizationDataSets
FXA_LOCALIZATION_SCRIPTS=${FXA_HOME}/data/localization/scripts
```

2.3.2 AWIPS variables (when using case data)

```
FXA_HOME=/awips/fxa
FXA_DATA=
FXA_CASE=/case
FXA_LOCAL_SITE=FSL
FXA_INGEST_SITE=FSL
FXA_NATL_CONFIG_DATA=${FXA_HOME}/data/localization
FXA_LOCALIZATION_ROOT=${FXA_HOME}/data/localizationDataSets
FXA_LOCALIZATION_SCRIPTS=${FXA_HOME}/data/localization/scripts
```

2.3.3 D3D variables

```
D3D_HOME=$D3D_DIR/app
D3D_SCR=$D3D_HOME/scripts
D3D_IMG=$D3D_HOME/bin/image
D3D_TCL=$D3D_HOME/lib/tcl/tclD3D
D3D_LOGS=$D3D_DIR/logs
D3D_HELP=$D3D_DIR/../docs/d3d_users_guide
D3D_PROC=$D3D_DIR/user_files
D3D_DATA=$D3D_DIR/localization/$FXA_LOCAL_SITE/data
D3D_TOPO=$D3D_DIR/localization/$FXA_LOCAL_SITE/topo
D3D_CONFIG=$D3D_DIR/localization/$FXA_LOCAL_SITE/config
```

2.4 Site specific data

The localization process provides site specific data to support operations at local offices. The localization for D3D *needs* match the AWIPS localization for your site. On the D3D CD both localizations happen to be **FSL**. If D3D is installed on a workstation with real-time AWIPS data and a localization other than **FSL**, the D3D localization should be updated to be compatible with real-time data.

2.4.1 D3D localization

To make a D3D localization compatible with an AWIPS localization using real-time data do one of the following, where **XYZ** is the three letter code for your site:

```
cd $D3D_DIR/localization        # change to the D3D localization directory
ln -s FSL XYZ                   # create a link for the local site that points to
                                # the FSL directory
```

or

```
cd $D3D_DIR/localization        # change to the D3D localization directory
rm d3denv                       # remove the existing link to FSL/env/d3denv
mv FSL XYZ                      # change the name of the FSL directory to that
                                # of the local site
ln -s XYZ/env/d3denv d3denv     # create a link for d3denv that points to d3denv
                                # in the local site’s directory
```

Once this has been completed, edit d3denv to replace the value of **FSL** with your site, **XYZ,** for both **FXA_LOCAL_SITE** and **FXA_INGEST_SITE**.

2.4.2 AWIPS Localization

If you choose to use the your AWIPS localization data no further steps are necessary. If you choose to use the AWIPS localization data from the D3D CD, one further step is required. Rename

```
$FXA_HOME/data/localizationDataSets/FSL
```

to

```
$FXA_HOME/data/localizationDataSets/XYZ
```

2.5 Basics

2.5.1 Prior to starting D3D

Configuring the window manager

Window managers have various methods for moving and resizing windows. Some performance degradation has been noted in D3D with certain methods, such as *Opaque* and *Transparent* movement methods (due to the way expose events are handled). The ***Box*** method of moving and resizing windows is recommended. These can be changed by launching your window manager’s configuration editor

Adjusting memory limits

The amount of memory that D3D will use is governed by the amount requested on the Vis5D command line when the Vis5D process is spawned. The code that spawns Vis5D is found in $D3D_dir/app/lib/tcl/tclD3D/fui_ipc/**fui_ipc.tcl**. Edit this file and find the line that contains “**set mbs [memory_size 200]**”. The default value is 200 (200 MB). Change this value to one that s appropriate. When this memory limit is reached, the least-recently-viewed graphics will be purged, thereby freeing the memory they used. When these graphics are needed again, they will be regenerated, causing other graphics to be purged. While D3D can be used under these conditions, performance may be inadequate.

2.5.2 Starting D3D

From the command line

To start D3D, execute the Bourne shell (sh) script, **app-d3d,** located in $D3D_DIR/app. This script will source the $D3D_DIR/localization/d3denv, set additional environment variables, and start the D3D application. Be sure that $D3D_DIR/app is in your path.

From an icon

An easy way to start an application is to set up an icon on your desktop that allows you to launch an application by clicking on it. Most Linux window managers have a *Launcher* allowing the user to customize their desktop environment. To set this up for D3D use the following suggestions with information specific to your installation

Name:	D3D_launcher
Comments:	Click twice to launch application
Command:	~awipsusr/d3d_demo/app/app-d3d
Type:	Application
Icon:	~awipsusr/d3d_demo/app/lib/tcl/tclD3D/icons_ppm/D3D_inon.ppm

2.5.3 Stopping D3D

To stop D3D, choose **Exit** under the File option on the Menu Bar. A dialog box appears in the middle of the display screen, asking you to confirm that you really want to exit.

If D3D crashes, for any reason, there is a script called $D3D_DIR/app/bin/**d3dkill** designed to clean up old files and kill legacy D3D processes. (D3D has been known to fail when reading data files that are not in the correct netCDF format, null, or corrupted).

Comments, Suggestions, or Questions

3.1 Contacts

3.1.1 Project Manager

Herb Grote	303.497.6110	grote@fsl.noaa.gov

3.1.2 Developers

Paula McCaslin	303.497.3187	mccaslin@fsl.noaa.gov
Phil McDonald	303.497.6055	mcdonald@fsl.noaa.gov
Ed Szoke	303.497.7395	szoke@fsl.noaa.gov